A Knowledge-Driven Approach to Activity Recognition in Smart Homes

Liming Chen, Chris D. Nugent, and Hui Wang

Abstract—This paper introduces a knowledge-driven approach to real-time, continuous activity recognition based on multisensor data streams in smart homes. The approach goes beyond the traditional data-centric methods for activity recognition in three ways. First, it makes extensive use of domain knowledge in the life cycle of activity recognition. Second, it uses ontologies for explicit context and activity modeling and representation. Third and finally, it exploits semantic reasoning and classification for activity inferencing, thus enabling both coarse-grained and fine-grained activity recognition. In this paper, we analyze the characteristics of smart homes and Activities of Daily Living (ADL) upon which we built both context and ADL ontologies. We present a generic system architecture for the proposed knowledge-driven approach and describe the underlying ontology-based recognition process. Special emphasis is placed on semantic subsumption reasoning algorithms for activity recognition. The proposed approach has been implemented in a function-rich software system, which was deployed in a smart home research laboratory. We evaluated the proposed approach and the developed system through extensive experiments involving a number of various ADL use scenarios. An average activity recognition rate of 94.44 percent was achieved and the average recognition runtime per recognition operation was measured as 2.5 seconds.

Index Terms—Activity recognition, activity ontologies, smart home, ontological modeling, semantic reasoning.

1 Introduction

With the rising aging population and overstretched healthcare resources, technology-driven healthcare delivery to support independent living has attracted increasing amounts of attention. Within this new paradigm, the concepts of Smart Homes (SH) have recently emerged as a viable mainstream approach to achieving this goal [1]. An SH is a residential home setting augmented with a diversity of multimodal sensors, actuators, and devices along with Information and Communication Technologies (ICT)-based services and systems [2]. By monitoring environmental changes and inhabitant's activities, an assistive system in an SH can process perceived sensor data, make timely decisions, and take appropriate actions to assist an inhabitant perform activities of daily living (ADL), thus extending the period of time living independently within their own home environment.

Currently, there are a number of SH projects being developed for the purpose of proof-of-concept demonstration in addition to the establishment of real living environments [3], [4]. There is a broad range of enabling technologies such as sensor networks, data communications and devices, that provide fragments of the necessary functionality required for the SH [5], [6]. This has led to, on the one hand, increasing capabilities of generating massive amounts of sensor data related to SH environments, inhabitants, and events, and on the other hand, the high expectation of providing novel advanced ADL recognition and assistance. Trends in the area of SH-based assistive living are now moving from location or time-based reminder systems or emergency-oriented reactive alert systems toward context-aware cognitive ADL assistance [7]. Cognitive ADL assistance intends to provide just-in-time activity guidance for elderly people and those suffering from, for example, cognitive deficiencies such as Alzheimer's disease, in completing their ADLs [8]. To achieve this objective, it is necessary to

1. monitor an inhabitant's behavior and their situated environment in real time,
2. dynamically fuse and interpret the multiple modalities of signals and features,
3. infer and recognize behaviors, changes or anomalies, continuously in real time in a progressive way, and
4. provide assistance to help the inhabitant perform the intended activity based on incrementally accumulated sensor data.

There is at present a major gap between the potential of data generation and the aspiration of advanced assistance provision in which context-aware personalized ADL assistance at multiple levels of granularity can be provided whenever needed. The central problem behind this gap is the lack of a novel, yet pragmatic, approach to activity recognition which is truly scalable and can be easily deployed within real living environments.

Activity recognition in an SH is presented with a number of challenges. First, ADLs can be carried out with a high degree of freedom in relation to the way and the sequential order they are performed. Individuals have different lifestyles, habits, or abilities and as such have their own way of performing ADLs. Though ADLs usually follow some kind of pattern, there are no strict constraints on the sequence and duration of the actions. For example, to prepare a meal one can first turn on the cooker and then place a saucepan

• *The authors are with the School of Computing and Mathematics, University of Ulster, Shore Road, Newtownabbey, County Antrim BT37 0QB, United Kingdom. E-mail: {l.chen, cd.nugent, h.wang}@ulster.ac.uk.*

Manuscript received 8 Apr. 2010; revised 9 Oct. 2010; accepted 22 Dec. 2010; published online 7 Feb. 2011.
Recommended for acceptance by Y. Chen.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2010-04-0208.
Digital Object Identifier no. 10.1109/TKDE.2011.51.

on the cooker, or vice versa. Such phenomena happen in almost all ADLs, e.g., preparing a drink, grooming, to name but a few. The wide range of ADLs and the variability and flexibility in the manner in which they can be performed require an approach that is scalable to large scale activity modeling and recognition.

Second, multimodal sensors coexist in an SH. They generate heterogeneous data different in both formats and semantics. It is often necessary to fuse and interpret sensor data from multiple sources in order to establish the context of the ongoing ADL. For instance, the ADL *makingtea* may involve the preparation of *tea bag*, *cup*, *hot water*, *milk*, and *sugar*. Only when some or all sensor data from these items are fused, can the ADL be recognized. In addition, sensor data are full of noises, e.g., missed activations and/or faulty readings. This increases the uncertainty of sensor data, as such the reliability of recognition.

Third, most ADLs are composed of a sequence of temporally related actions. As such, sensor data related to an ADL are generated incrementally as the ADL unfolds. In order to provide context-aware assistance for an SH inhabitant, activity recognition should be performed at discrete time points in real time in a progressive manner. This will accommodate the ever-changing sensor data to recognize the current state of the ongoing activity and further identify the user's needs at the correct time.

Current researches on activity recognition have mainly focused on the use of probabilistic and statistical analysis methods, the so-called data-driven approach, for single-user single-activity scenarios [17], [18], [19], [20]. In this paper, we present a knowledge-driven approach to the processing of multisource sensor data streams for the purposes of activity recognition. The purpose of our study is not to extend existing data-mining methods to address complex activity scenarios, e.g., interleaved or concurrent activities, sensor noise-caused uncertainty, and multioccupancy, however, to develop an alternative activity recognition paradigm that can address the aforementioned challenges. The approach is motivated by the observations that ADLs are daily routines full of commonsense knowledge providing rich links between the environment, events, and activities. In addition, user profiles, i.e., an inhabitant's ADL preferences and their specific ways of performing ADLs, provide prior personal-level details about the ADL itself. Such domain and prior knowledge is valuable in creating ADL models, avoiding the need of large-scale data set collection and training.

Central to the approach is the formal explicit ontological modeling and representation of the SH domain, i.e., SH context and ADLs. Ontological activity modeling provides a description-based modeling method. It models activities as a hierarchy of classes with each class described by a number of properties. As such, the generated activity models are able to capture built-in interrelations between objects and activities irrelevant of the sequence in which these objects are used. Context ontologies serve as a situation model that interlinks multisource sensor data to build a situation at specific time points. These situations can then be interpreted in terms of the ADL models to infer activities. ADL ontologies can be used as a seed ADL classification model. The seed model can, on the one hand, be directly used to interpret situations for activity recognition, and on the other hand, grow naturally by learning undefined activity patterns from ongoing data mining. With the proposed approach presented in this paper, activity recognition is equivalent to performing subsumption reasoning using dynamically constructed SH situations against ADL descriptions. Activity assistance is reduced to instance checking in ontological reasoning, i.e., to discover the most closely matched activity profile for each user with regard to the recognized activity and subsequently to use the missing properties for assistance provision.

We identify three main areas where contributions from the current study have been made. First, we propose a knowledge-driven approach to activity recognition for the SH. The approach addresses the difficulty of activity modeling caused by the diversity of ADLs and the flexibility of performing ADLs, by providing unified explicit ontological activity models. Ontological ADL modeling can model ADLs as generic activity structures, i.e., the terminologies of ADL ontologies, and specific user ADL profiles, i.e., the assertions of ADL ontologies, thus facilitating the exploitation of domain knowledge at different levels of granularity. It addresses the heterogeneity and fusion of multiple sensor data through descriptive context modeling, i.e., through the use of context ontologies. Second, we develop a novel algorithm for activity recognition based on the subsumption reasoning of description logic (DL). The compelling feature of the recognition algorithm is that it can organize the recognition results into a hierarchy of activities at different levels of abstraction at a specific point in time. Given that activity recognition proceeds in a progressive manner, the recognized activities will be gradually narrowed down from high-level abstract activities to low-level specific activities and finally to a concrete activity. As such the algorithm can support both coarse-grained and fine-grained activity recognition. Third, we implement a feature-rich system for the proposed approach and deploy it in our SH research laboratory. The system is tested by actors with real sensor deployments and use cases and evaluated by a number of metrics. Comparing to existing data-driven approaches that are theoretically also able to carry out real-time online activity recognition, our approach is technically applicable and practically scalable to real-world scenarios involving large-scale multiple activities. In addition, the technologies and implementation not only demonstrate the benefits of knowledge-driven activity recognition but also provide an infrastructure for applying this approach to other use cases beyond the realms of AAL. To do this, only requires the provision of domain-dependent and application-specific activity and context models, i.e., ontologies. As such, the approach is applicable to a wide range of application contexts such as intelligent meeting rooms [9], [10].

The remainder of the paper is organized as follows: Section 2 discusses related work. Section 3 analyzes the SH domain and presents ontological context and ADL modeling. Section 4 outlines the system architecture for the proposed approach. Section 5 describes activity recognition algorithms. We discuss implementation, testing, and evaluation in Section 6 and conclude the paper in Section 7.

2 Related Work

Activity recognition approaches can be generally classified into two categories. The first is based on the use of visual sensing facilities, e.g., camera-based surveillance systems, to monitor an actor's behavior and environmental changes [11], [12]. The approaches in this category exploit computer vision techniques to analyze visual observations for pattern recognition [13], [14]. The second category is based on the use of emerging sensor network technologies for activity monitoring. The sensor data are analyzed using data mining and machine learning techniques to build activity models, which are then used as the basis of activity recognition. In these approaches, sensors can be attached to an actor under observation—namely wearable sensors, or objects that constitute the activity environment—namely dense sensing. Wearable sensors often use inertial measurement units and RFID tags to gather an actor's behavioral information [15]. This approach is effective for recognizing physical movements such as physical exercises [16]. In contrast, dense sensing infers activities by monitoring human-object interactions [8] through the usage of low-power and low-cost sensors.

With the dense sensing paradigm, activity recognition can be performed in several strands with the fundamental differences relating to the manner in which activities and an inhabitant's ADL profile are modeled and represented. One strand is referred to as the generative approach, which attempts to build a complete description of the input or data space, usually with probabilistic analysis methods such as Markov models [17] and Bayesian networks [18] for activity modeling. These methods incorporate an inhabitant's preferences by tuning the initial values of the parameters of the probabilistic models. The major disadvantage with such methods is that the model is static and subjective in terms of probabilistic variable configuration. An alternative approach is referred to as the discriminative approach, which only models the mapping from inputs (data) to outputs (activity labels). Discriminative approaches include many heuristic (rule-based) approaches, for example, neural networks, linear, or nonlinear discriminant learning. They use machine learning techniques to extract ADL patterns from observed daily activities, and later use the patterns as predictive models [19], [20]. Both approaches require large data sets for training models, thus suffer from the data scarcity or the "Cold Start" problem. It is also difficult to apply modeling and learning results from one person to another.

The third strand, i.e., the logical approach, uses logical formalisms, for example, event calculus [7] and lattice theory [21], for representing ADL models and conducts activity explanation and predication through deduction or abduction reasoning. Comparing to the above two data-centric approaches, logical approaches are semantically clear in modeling and representation and elegant in inference and reasoning.

Our approach is based on ontological activity modeling and representation. It is closer to the logical approach in nature in that it uses a Description Logic-based markup language (i.e., OWL and RDF (www.w3.org)) for specifying conceptual structures and relationships. Both languages support inference and reasoning. The compelling feature of our approach is that it can model domain knowledge at two levels of abstraction. Common, generic activity knowledge can be modeled at the conceptual level as an activity class described by a number of properties. These properties describe the types of objects that can be used to perform the activity. In this way, activity models can be created without the requirement of large amounts of observation data and training processes. They are applicable and reusable to a wide range of users. On the other hand, the specific way of a user performing an activity can be modeled as an instance of a corresponding conceptual activity model. The instance will consist of specific information related to how the activity is performed, e.g., objects used and the sequential order they are used in. This provides a mechanism for accommodating the personal nature of individual user's ADLs. The generated user-specific activity models, also referred to as user profiles, can later be used for realization of personalized ADL assistance.

Given that most ADLs are daily routines with abundant prior knowledge of their likely patterns stemming from medical observations and psychological behavioral studies [23], [24], it is possible to manually construct conceptual activity models, i.e., ADL ontologies using existing ontology engineering-based tools. The creation of user activity profiles is then equivalent to creating activity instances in terms of a user's preference and their manner of performing ADLs. Hence, it is straightforward to undertake and also scalable to a larger number of users and activities in comparison with traditional approaches.

Ontologies have been previously used for both vision-based [22], [25] and dense sensing-based activity recognition [26], [27]. Nevertheless, their major purpose in existing work has been to provide common terms as building primitives for activity definitions as in [24] or to address model incompleteness and multiple representations of terms as in [26], [27]. Activity recognition is still based on data-centric probabilistic methods or machine learning techniques such as Markov Models and Bayesian networks. In comparison, the proposed knowledge driven approach uses ontological modeling, representation, and reasoning for sensor modeling, context fusion, activity modeling and recognition covering the life cycle of the process. This brings two additional benefits. First, ontological contextual modeling facilitates sensor data exchange and reuse between intra- and interorganizations. Second, machine understandable and processable semantic sensor data allows automated interpretation and reasoning, thus enabling higher levels of automation. We shall discuss the details further in the following sections.

3 Ontological Modeling for the SH Domain

Inhabitants in an SH usually perform routine daily activities in specific circumstances, i.e., in specific locations with specific objects at specific times. For example, brushing teeth normally takes place twice a day, in the bathroom, in the morning and before going to bed. It usually involves the use of toothpaste, a toothbrush, and water. This is generally referred to as the context for the corresponding activity. As humans have different lifestyles, habits, or abilities, an individual's ADLs and the manner in which they perform

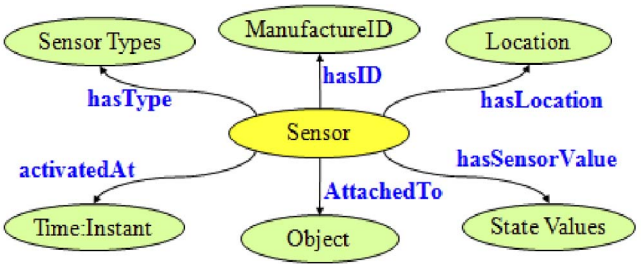

Fig. 1. The generic conceptual sensor model.

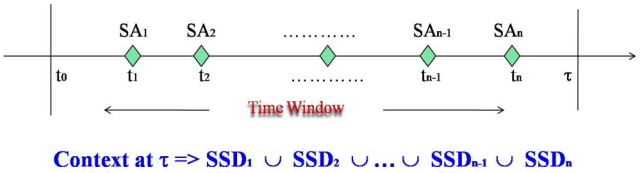

Fig. 2. The illustration of the situation formation process. $\mathrm{SA}n$ denotes a sensor activation at time point tn and $\mathrm{SSD}n$ denotes the semantic descriptions of the activated sensor at $\mathrm{t}n$.

them may vary from one person to another. For instance, one person may prefer white coffee and another person may prefer black coffee. Even for the same type of activity, e.g., making white coffee, different people may use different items, e.g., skimmed milk or whole milk, and complete the task in a different order, e.g., adding milk first and then sugar, or vice versa. As such, ADLs can be categorized as *generic* ADLs applicable to all and *personalized* ADLs taking into account individual subtleties. In addition, ADLs can be conceptualized at different levels of granularity. For example, Grooming can be considered to be comprised of the subactivities *Washing*, *Brushing*, and *Applying Makeup*. There are usually a "*is-a*" and "*part-of*" relationship between a primitive and composite ADLs. All these observations could be viewed as prior domain knowledge that can facilitate activity modeling and recognition.

Ontological modeling is the process to explicitly specify key concepts and their properties for a problem domain. These concepts are organized in a hierarchical structure in terms of their shared properties to form superclass and subclass relations. For example, *MakeTea* is a subclass of *MakeHotDrink*. Properties establish the interrelations between concepts. For instance, *hasDrinkType* is a property of the *MakeHotDrink* activity that links the *DrinkType* concept (e.g., tea, coffee, chocolate) to the *MakeHotDrink* concept. Both concepts and properties are modeled using the commonly shared terms in the problem community. The resulting ontologies are essentially knowledge models able to encode and represent domain knowledge and heuristics. This avoids the manual class labeling, preprocessing, and training processes in the traditional approaches to activity recognition. In addition, ontologies allow software agents to interpret data and reason against ontological contexts, thus enhancing the capabilities of automated data interpretation and inference.

3.1 Ontological Context Modeling

SH inhabitants perform ADLs in a diversity of temporal, spatial, environmental contexts. Spatial contexts relate to location information and surrounding entities such as rooms, household furniture, and appliances. Event contexts contain background activities and dynamic state changes of appliances and devices. Example events could be the state changes of doors, windows, lights, alarms, a cooker, and taps. Environmental contexts are composed of environmental information such as temperature, humidity, and general weather conditions. Temporal contexts indicate the time and/or duration. There is a high correlation between ADLs and contexts. For example, a cooking ADL happens in the kitchen with a cooker turned on. A grooming ADL takes place in the bathroom in the morning.

Contextual information is usually captured through various sensors. Each sensor monitors and reflects one facet of a situation. Based on this observation, our context modeling is centered on ontological sensor modeling. As can be viewed from the generic conceptual model in Fig. 1, sensors are inherently linked to a number of physical and conceptual entities such as objects, locations, and states. For example, a contact sensor is attached to a teapot in the second cupboard to the left of the sink in the kitchen. By explicitly capturing and encoding such domain knowledge in a sensor model, it is possible to infer the corresponding objects and location from the activation of the sensor. This implies that an inhabitant performs an activity in the inferred location with the inferred object.

As most ADLs require the interlinking and fusion of data from multiple, disparate sensor sources in order to infer the high-level activities, it is necessary to aggregate a sequence of sensor activations to generate a situation at a specific time point. Fig. 2 shows the situation formation process, which can be described as follows: Each sensor has a default state value for its state property, denoting the state of the object to which it is attached. When a sensor is activated, the state property will change. Subsequently, the system will translate the state change as an occurrence of a user-object interaction at the specific time. As it is difficult, if not impossible, to monitor what happens at detailed levels after an object is interacted with, it is common practice to interpret a sensor activation as a user-object interaction, ignoring how the object is used and when it is deactivated. As such, a user-object interaction is equivalent to an instantaneous sensor activation and can be interpreted as an object has been used for performing an activity. In this way, by aggregating individual user-object interactions along a timeline (refer to Fig. 2) the situation at specific time points can be generated.

The conceptual sensor model depicted in Fig. 1 can be formally represented in the SH context ontologies—refer to Figs. 3a and 3b. Context ontologies consist of classes and properties for describing SH entities such as *Device*, *Furniture*, *Location*, *Time*, and *Sensor*, along with their interrelationships.

3.2 Ontological ADL Modeling

In order to perform activity recognition, computational activity models are required. Based on the nature and characteristics of ADLs, we developed a description-based conceptual activity model as presented in Fig. 4. In addition to the name and textual description, an activity can be described by a number of properties. These properties relate

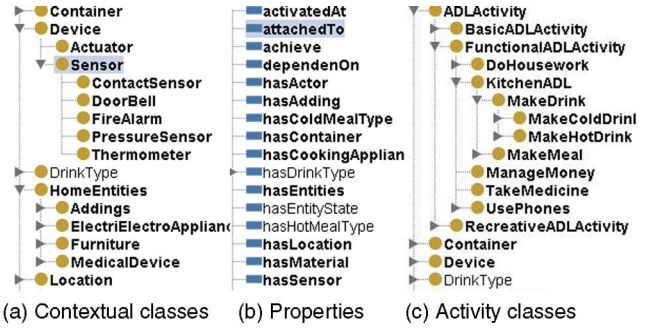

Fig. 3. A fragment of the SH domain ontologies.

an activity to other physical items and conceptual entities. They can be categorized into three groups. The first group represents the context, e.g., time, location and actors, within which the activity takes place. The second group represents the causal and/or functional relations, e.g., conditions and effects that are used for inference during high-level activity reasoning. The properties in the third group denote the type and interrelationship between activities. With the diversity of ADLs in a SH, this conceptual model will serve as a base model and can be extended to cover ADLs at multiple levels of abstraction.

The conceptual activity model depicted in Fig. 4 can be structured and represented in formal ADL ontologies in Fig. 3c. The ADL ontology consists of an activity hierarchy in which each node, also called a class, denotes a type of ADL. Each class is described with a number of properties. A property is defined by specifying its domain and range. The domain refers to all classes that can be described by the property and the range refers to all classes whose instances can be assigned to the property. A property describes a class using either a literal or an instance of another class as its value, thus linking two classes. Subclasses can inherit all properties from its superclass. This can be illustrated using the previous example, i.e., *hasDrinkType* is a property. Its domain is the *MakeHotDrink* activity class and its range the *DrinkType* class. The instances of the *DrinkType* class, e.g., tea and coffee, are the values that the *hasDrinkType* property will take.

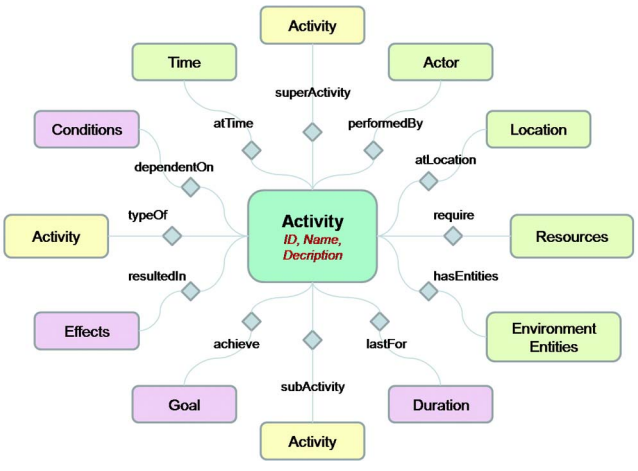

Fig. 4. The conceptual activity model.

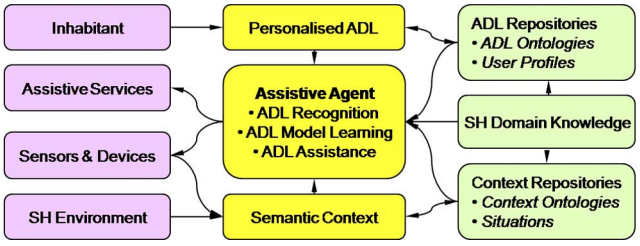

Fig. 5. Knowledge-driven activity recognition architecture.

In the proposed knowledge-driven approach, ADL ontologies can be viewed as activity models that establish links between activities and contextual information through activity-based properties. Context ontologies can be regarded as feature models for constructing situations at specific time points that link contextual information with sensor observations through context properties. As such, the whole process of assisted living ranging from low-level sensor data collection, middle-level data fusion, to high-level activity recognition can be streamlined in a unified modeling, representation and reasoning formalism. Activity recognition amounts to constructing situations from sensor observations and reasoning them against activity models. In the following section, we shall describe a system architecture and the corresponding activity recognition algorithms for the proposed approach.

4 The System Architecture

Fig. 5 depicts the proposed system architecture for the knowledge-driven activity recognition in SH. Central to the architecture is the ontological modeling and representation of ADLs and SH contexts. This not only avoids the fundamental difficulty of obtaining labeled data to learn activity models but also provides an effective way to incorporate domain knowledge into both context and activity models. As can be viewed in Fig. 5, the ADL Repositories component consists of ADL models at two levels of abstraction, i.e., ADL ontologies as the generic ADL models and User Profiles as a user's personalized ADL model. A user's ADL profile is in essence an instance of the corresponding ADL classes in the ADL ontologies with specific binding in terms of the user's activity preferences. Similarly, the Context Repositories component contains conceptual context models, i.e., the context ontologies, which are used to instantiate sensor observations to create situations at discrete time points. A situation at a specific time point is, on one hand, used for activity recognition in real time. On the other hand, it can be archived over a longer period of time and used to extract ADL patterns or detect changes in the way the ADL is being completed. For example, a user regularly makes tea over a period of three months at a specific time in the day, or the duration for making tea increases over a given period of time.

The components in the left-hand column of Fig. 5 denote the physical environment of the SH, which consists of users, sensors, actuators, and assistive services. The sensors are responsible for monitoring environmental, event, and activity contexts. Assistive Services receive instructions from the ADL Assistive Agent and act on the environment,

devices, and/or the inhabitant through various actuators. Many research issues relating to the hardware aspects in an SH, e.g., optimal deployment of sensors, still remain unsolved, however, are beyond the scope of this paper.

The Assistive Agent is the core component of the system, which takes as inputs the semantic descriptions of a situation and performs activity recognition with regard to the ADL models in the ADL Repositories. A situation at a specific time is generated by aggregating user-object interactions as described in Section 3.1. Activity recognition makes use of description-based reasoning capabilities enabled by ontological modeling and representation. The agent performs coarse-grained activity recognition by reasoning semantic situations against generic activity models, i.e., ADL ontologies, and fine-grained activity recognition against a user's ADL profile. This allows an assistive agent to provide generic and personalized assistance, respectively, in terms of a user's need and the level of details of recognized activities. In addition, given the diversity of ADLs and discrepancy of an individual's living styles and preferences, the manually built ADL ontologies will be treated as the seed of the ADL models. When a large amount of sensor data pertaining to an inhabitant's ADLs are captured and mined over a long period of time, regular activities that have not been modeled before, either a new generic activity or a personalized one with more subtlety can then be identified and extracted. The agent can use these learned activities to grow ADL ontologies. As such, activity models can evolve, and subsequently improve the performance of activity recognition. Details relating to activity model learning, which are not covered here due to limited space, can be found in [28].

5 Activity Recognition

To help illustrate our approach, we use the *MakeDrink* ADL class hierarchy, as shown in Fig. 3, as an example for discussion. An activity model in the ADL ontologies is a concept described by a number of properties that specify relationships between the activity and other entities. Specifically, the *MakeDrink* activity is described with two inherited properties *hasActor* and *hasLocation*, and two specific properties *hasContainer* and *hasAddings*. Its subclasses, e.g., *MakeHotDrink* and *MakeColdDrink* have additional properties, e.g., drink types. An inhabitant's ADL profile is a set of ADL instances defined by incorporating the user's preferences and life styles. For instance, an inhabitant ADL profile for *MakeDrink* may contain a *MakeHotDrink* instance with properties *hasActor(theInhabitant), hasLocation(kitchen), hasContainer(cup), hasHotDrinkType(coffee), and hasAddings(semiskimmedMilk)*.

5.1 The Theoretical Foundation

We use the Web Ontology Language (OWL) for ontological modeling and representation. OWL is theoretically based on the well-developed knowledge representation formalism of Description Logic [29]. As such, our proposed approach to activity recognition is built upon DL theories and reasoning mechanisms, which are described below.

The core elements of the DL formalism are concepts, roles, and individuals. Concepts denote sets of individuals.

TABLE 1
Concept Formation Syntax and Element Notations

Formation Syntax	Some Element Notations
$\mathcal{C}, \mathcal{D} \rightarrow$	
$\mathcal{A} \mid$ (atomic concept)	$\mathcal{C}1 \sqsubseteq \mathcal{C}2$ (subclass of)
$\top \mid$ (universal concept)	$\mathcal{C}1 \equiv ... \equiv \mathcal{C}n$ (equivalent class)
$\bot \mid$ (bottom concept)	$\mathcal{R}1 \sqsubseteq \mathcal{R}2$ (subproperty of)
$\neg\mathcal{A} \mid$ (atomic negation)	$\mathcal{R}1 \equiv ... \equiv \mathcal{R}n$ (equivalent class)
$\mathcal{C} \sqcap \mathcal{D} \mid$ (intersection)	$\mathcal{O}1 = ... = \mathcal{O}n$ (same individual)
$\forall\mathcal{R}.\mathcal{C} \mid$ (all value restriction)	$\mathcal{C}i \sqsubseteq \neg\mathcal{C}j$ (disjoint classes)
$\exists\mathcal{R}.\mathcal{C} \mid$ (some value restriction)	$\mathcal{O}i \neq \mathcal{O}j$ (different individuals)
$\{\mathcal{O}1,...\mathcal{O}n\} \mid$ (enumeration)	$\mathcal{R}1 \equiv \mathcal{R}2^{-}$ (inverse of)
$\exists\mathcal{R}.\{\mathcal{O}\} \mid$ (property value)	$\mathcal{R}^{+} \sqsubseteq \mathcal{R}$ (transitive)
$\geq n\mathcal{R}.\mathcal{C}, \leq n\mathcal{R}.\mathcal{C}, = n\mathcal{R}.\mathcal{C} \mid$ (min, max, cardinality)	$\mathcal{R} \equiv \mathcal{R}^{-}$ (symmetric)

Roles denote binary relationships between individuals. Individuals are instances of concepts. The vocabulary used for defining concepts and roles of an application domain is referred to as the terminology or the *TBox* in short. All named individuals in terms of vocabulary are referred to as assertions about a real world domain or the *ABox*. In addition to atomic concepts and roles in the *TBox*, all DL systems allow users to build complex descriptions of concepts and roles. This can be performed using the syntax and constructors of description languages. As DL has a model-theoretic semantics, the statements in the *TBox* and the *ABox* can be interpreted with rules and axioms in DL, thus enabling reasoning and inference, e.g., subsumption and satisfiability reasoning [30].

Consider, in abstract notation, we use the letter $\mathcal{A}$ for atomic concepts, the letter $\mathcal{R}$ for atomic roles, the letter $\mathcal{T}$ for *TBox*, and the letters $\mathcal{C}$ and $\mathcal{D}$ for concept descriptions. Concept descriptions in OWL can be formed using the basic elements in Table 1.

DL supports a number of reasoning tasks [30], [31], including satisfiability, subsumption, equivalence, disjointness, and consistency. Satisfiability is to check whether a newly defined concept makes sense or whether it is contradictory. The following three tasks are to infer relationships between concepts. For example, subsumption is to find out whether a concept is more general than another one. A concept $\mathcal{C}$ is subsumed by a concept $\mathcal{D}$ if the set of individuals denoted by $\mathcal{C}$ is a subset of the set denoted by $\mathcal{D}$. Actually, all inferences about concept interrelationships can be reduced to satisfiability reasoning.

DL reasoning has been well studied, which supports decidability, completeness, and soundness in polynomial time complexity for an inexpressive DL and in exponential time complexity for expressive DLs [29], [31]. With the advanced tableau algorithms and various optimization techniques, modern reasoners such as FACT++, Pellet and Racer have substantially improved the performance in terms of not only a quantitative change but also a qualitative change from an exponential growth in solution time to an almost constant solution time growth [32]. Current semantic data infrastructures can support semantic classification and queries in seconds against a knowledge base of millions of triples (http://challenge.semanticweb.org). This provides

sufficient technological support for our knowledge-driven approach.

With ontological modeling, activities are modeled as activity classes in the ADL ontologies and contextual information such as location and SH objects are modeled as properties for describing activity classes. As such, a situation at a specific time point is actually a concept description created from SH contextual ontologies, denoting an unknown activity. In this case, activity recognition can be mapped to the classification of the unknown activity into the correct position of the class hierarchy of the activity ontologies and the identification of the equivalent activity class. This is the subsumption problem in $\mathcal{DL}$, i.e., to decide if a concept description $\mathcal{C}$ is subsumed by a concept description $\mathcal{D}$, denoted as $\mathcal{C} \sqsubseteq \mathcal{D}$. The commonly used tableau proof system uses negation to reduce subsumption to unsatisfiability of concept descriptions, which can be described below

- *Reduce subsumption to check unsatisfiability of concept description, i.e., a concept $\mathcal{C}$ is subsumed by a concept $\mathcal{D}$ can be reduced to the checking of satisfiability of concept $\mathcal{C}$ and the negation of concept $\mathcal{D}$, which can be written below*

$$\mathcal{C} \sqsubseteq \mathcal{D} \mapsto \mathcal{C} \cap \neg\mathcal{D}.$$

- *Check whether an instance b of this resulting concept description can be constructed.*
- *Build a tree-like model for the concept description.*
- *Transform the concept description in Negation Normal Form.*
- *Decompose the description using tableau transformation rules.*
- *Stop when a clash occurs or no more rules are applicable.*
- *If each branch in the tableau contains a clash, the concept is inconsistent.*

Specifically a situation, i.e., an unknown concept description at a specific time point can be generated by linking sensor observations to properties of the context ontologies (as shown in Fig. 1) and incrementally fusing a sequence of sensor observations (as shown in Fig. 2). For example, the activation of the contact sensors on a cup and milk bottle can link the *cup* and *milk* to the unknown activity through *hasContainer* and *hasAddings* properties. By aggregating sensor observations along a timeline, a specific situation, that corresponds to an unknown activity, could be reached, e.g., *hasTime*(*10am*), *hasLocation*(*kitchen*), *hasContainer*(*cup*), and *hasAddings*(*milk*). If the closest ADL class in the ADL ontologies that contains as many perceived properties as possible to the situation can be found, e.g., *MakeDrink*, then it can be deemed to be the type of ADL for the identified situation.

5.2 Activity Recognition Method

Suppose that we create SH-based knowledge repositories $\mathcal{KR}(\mathcal{T}, \mathcal{A})$ as described in Section 3, which consist of a set of terminological axioms $\mathcal{T}$, i.e., ontological concepts and a set of assertional axioms $\mathcal{A}$, i.e., instantiated facts (instances). The activity recognition algorithm is depicted in Fig. 6 and described as follows:

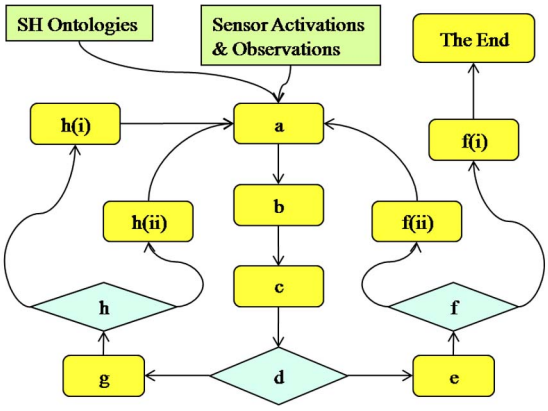

Fig. 6. The activity recognition algorithm.

1. Detect sensor activations and convert them to corresponding ADL properties in the ontologies.
2. Use context ontologies to aggregate and fuse multiple sensor observations to construct a situation at individual time points.
3. Construct an activity description, denoted as $\mathcal{ATV}$, at two levels of abstraction. $\mathcal{ATV}$-$\mathcal{C}$ denotes the conceptual description of $\mathcal{ATV}$ whereas $\mathcal{ATV}$-$\mathcal{I}$ denotes its instance that binds properties with sensor readings.
4. Perform equivalency and subsumption reasoning to check whether $\mathcal{ATV}$-$\mathcal{C}$ is equivalent to any atomic concept $\mathcal{REC}$-$\mathcal{ATV}$ in $\mathcal{T}$. If that is the case, go to step e, otherwise go to step g.
5. If $\mathcal{ATV}$-$\mathcal{C}$ is equivalent to an atomic activity concept $\mathcal{REC}$-$\mathcal{ATV}$, then we can recognize $\mathcal{REC}$-$\mathcal{ATV}$ as the type of the underlying activity. We still need to decide whether it is an abstract activity such as *MakeDrink* or a specific activity, e.g., *MakeTea*.
6. Use semantic retrieval to obtain the set of atomic activity concepts $\mathcal{SUB}$-$\mathcal{ATV}$-$\mathcal{SET}$ in $\mathcal{T}$ subsumed by $\mathcal{REC}$-$\mathcal{ATV}$. These are the equivalents and descendants of $\mathcal{REC}$-$\mathcal{ATV}$ in $\mathcal{T}$
   a. If $\mathcal{SUB}$-$\mathcal{ATV}$-$\mathcal{SET}$ is empty, this means $\mathcal{REC}$-$\mathcal{ATV}$ has no subactivity concepts and $\mathcal{REC}$-$\mathcal{ATV}$ is a specific activity, e.g., *MakeTea*.
   b. If $\mathcal{SUB}$-$\mathcal{ATV}$-$\mathcal{SET}$ is not empty, this means $\mathcal{REC}$-$\mathcal{ATV}$ has subactivity concepts and $\mathcal{REC}$-$\mathcal{ATV}$ is an abstract activity, e.g., *MakeDrink*. In this case, a general activity can be recognized such as *MakeDrink*. It will need further sensor data to decide which subactivity the $\mathcal{ATV}$-$\mathcal{C}$ is, e.g., to decide if it is *MakeColdDrink* or *MakeHotDrink*. This allows incremental activity recognition.
7. If $\mathcal{ATV}$-$\mathcal{C}$ is not equivalent to any atomic activity concept, use semantic retrieval to obtain the most specific atomic concepts $\mathcal{MSC}$-$\mathcal{ATV}$ in T subsuming $\mathcal{ATV}$-$\mathcal{C}$. In essence, the $\mathcal{MSC}$-$\mathcal{ATV}$ is the direct superconcept of $\mathcal{ATV}$-$\mathcal{C}$.
8. Use semantic retrieval to obtain the set of atomic activity concepts $\mathcal{SUB}$-$\mathcal{ATV}$-$\mathcal{SET}$ in $\mathcal{T}$ subsumed by $\mathcal{MSC}$-$\mathcal{ATV}$. These are the equivalents and descendants of $\mathcal{MSC}$-$\mathcal{ATV}$ in $\mathcal{T}$

a. If $\mathcal{SUB}$-$\mathcal{ATV}$-$\mathcal{SET}$ is empty, this means $\mathcal{MSC}$-$\mathcal{ATV}$ has no subactivity concepts and $\mathcal{MSC}$-$\mathcal{ATV}$ is a leaf activity, e.g., *MakeTea*. But the activity has not completed. This means the approach can recognize activities with incomplete sensor data.
b. If $\mathcal{SUB}$-$\mathcal{ATV}$-$\mathcal{SET}$ is not empty, the activity recognition process will be similar to the case in 6b with $\mathcal{MSC}$-$\mathcal{ATV}$ replacing $\mathcal{REC}$-$\mathcal{ATV}$.

It is worth pointing out that the above algorithm describes the process of a single round of activity recognition, i.e., given an activity description (equivalent of a set of sensor data) discover the possible activities. This process could be repeated many times in order to realize continuous progressive activity recognition—see Section 5.3 for details. To illustrate the algorithm, assume that a kitchen door sensor has activated and further the cup is only used for making drink. Suppose that the contact sensor attached to a cup is activated. This means that the *cup*, as an instance of Container, is used in an ADL. As the Container class is the filler of *hasContainer* property, it can be inferred that the *hasContainer* property is assigned the value *cup*. Since the *hasContainer* property is used to describe the *MakeDrink* class, it can then be inferred that a *MakeDrink* ADL has taken place. Though it is not possible to ascertain whether the ADL is *MakeHotDrink* or *MakeColdDrink* as both ADLs have the *hasContainer* property, nevertheless, based on the limited sensor information the agent can identify the high-level ADLs, i.e., the inhabitant is performing a *MakeDrink* ADL. If, as the ADL unfolds, we obtain sensor data about the use of *coffee*, then we can determine that the inhabitant is making a hot drink. From the above discussion, it is apparent that the proposed approach can monitor the unfolding of an ADL and continuously recognize the ultimate ADL with an increasing level of accuracy and certainty, which may be considered as not being possible with other approaches.

5.3 Real-Time, Continuous Activity Recognition

Activity classes in ADL ontologies are structured in a hierarchical tree with subclasses inheriting all properties from their superclasses. The closer to the leaf of the class tree the more properties with which the activity is described, and the more specific the activity is. When an ADL is performed in the real world along the temporal dimension, the contextual information related to the ADL will be captured incrementally. With less contextual information, e.g., at the initial stage of the ADL, subsumption reasoning can only classify an ADL to a generic activity class. Nevertheless, as the ADL unfolds, more contextual information will become available to enable the creation of an increasingly rich activity description. As such, by performing ADL subsumption reasoning dynamically along a timeline, as shown in Fig. 2, it is possible to recognize an ongoing ADL continuously and progressively.

As ontological activity models are description based, i.e., based on the values or status of their properties, they do not model temporal aspects explicitly in the activity models themselves. Nevertheless, as the status of a property is coupled with sensor activation and a sensor activation is time stamped (refer to Section 3.1), temporal data are actually implicitly embedded in the occurrence of a user-object interaction. As such, the handling of the temporal aspects is carried out at system operational level, which is described in the following paragraphs.

Two methods are used to handle temporal reasoning in order to support real-time, continuous activity recognition. The first is the sliding time window technique for sensor activation fusion. A time window is a fixed duration of time within which all sensor activations are aggregated to generate an activity description. The generated description serves as the input to the subsumption reasoner for activity recognition. If an activity is successfully recognized at a time point within the time window, e.g., the third minute, the algorithm will clear all activated sensors accumulated so far and the time window will restart from the time point (e.g., the third minute) again, i.e., the time window is sliding each time an activity is recognized. The sliding window technique provides a mechanism for activity partitioning along a timeline and also a method to decide which sensors should be discarded and which should be aggregated to form an activity description. If the algorithm cannot recognize an activity within the time window, the system will deem that the activity is aborted. It will then abort this round of recognition reasoning, i.e., clear all existing activated sensors and restart sensor monitoring and time window again. Note that within the time window if the system does not receive any user interactions within a fixed amount of time after the last sensor activation it will provide action reminders for the user—i.e., providing activity assistance. As such, the system only aborts activity recognition after it has attempted to help the user and the user does not respond. A possible scenario is that the system can send an alarm if it infers that the user should do something, however, fails to do so. This is the assistance aspect of the system we do not cover in this paper.

The length of the sliding window is determined as follows: A fixed maximum amount of time is provided as the default duration of the sliding time window. In addition, the duration of an activity is modeled as a duration property in its ontological model. The value of the duration property can be initially decided based on domain knowledge and later refined through learning. As such, the sliding time window can be adjusted once an activity is initially recognized. For example, a *MakeDrink* activity may set its duration property as 4 minutes and a *MakeMeal* activity as 15 minutes, respectively. If they are recognized in the continuous progressive activity recognition, the time window can be reset to the corresponding value. In this way, the correct set of sensor activations can be guaranteed to be used for activity description generation, thus improving recognition accuracy.

The second method for handling temporal reasoning is to determine when a recognition operation is performed, i.e., the activity description from sensor fusion within the time window is fed into the underlying subsumption reasoner for activity recognition. Activity recognition in assistive living is different from activity recognition in traditional data mining where a prior data set is available and recognition can be performed offline. In assistive living, an activity must be continuously recognized in real time to detect anomalies

and difficulties during the performance of the user in order to provide just-in-time assistance. This requires the recognition reasoning as described in Section 5.2 to be performed repeatedly as the activity unfolds. A recognition operation can be initiated in two ways. The first is to specify a fixed time interval (a frequency), i.e., a recognition operation is performed periodically when the time interval elapses. The second way is to use the sensor activation as a trigger, i.e., each time a user-interaction happens (i.e., an action has been performed) the recognition operation is performed. With the ontological activity models, the sensor activation and aggregation models and the two temporal handling methods the system is able to support real-time continuous activity recognition.

Another feature of the algorithm is that it can offer both coarse-grained and fine-grained activity recognition. The former is based on subsumption reasoning against *TBox*, i.e., concept descriptions at the terminological level. In this case, following the activity recognition algorithm in Section 5.2, an extra step may be to compare the properties of the recognized activity with the properties identified by sensor observations. The missing properties can then be used to suggest the next action(s). For example, if the *MakeTea* activity is described by properties *hasContainer*, *hasAddings*, and *hasHotDrinkType*, and the sensors observe *tea* as *HotDrinkType* and *cup* as *Container*, then advice on *Addings* such as milk or sugar can be provided.

Fine-grained activity recognition and assistance is based on subsumption reasoning against *ABox*, i.e., a user's ADL profiles at the instance level. In this case, once the type of activity that an inhabitant performs (as described in Section 5.1) is recognized, the instances of the inhabitant performing the type of activity in terms of his/her ADL profile can be retrieved from the $\mathcal{KR}(\mathcal{T}, \mathcal{A})$. The discovered ADL instance can then be analyzed in terms of sensor observations. Fine-grained activity recognition and assistance will not only recommend the types of action to be performed, but also the items/objects used for the action. For example, suppose an unknown ADL has been identified in the context of *hasContainer(cup)*, *hasAddings(milk)*, and *hasHotDrinkType(coffee)*, using the aforementioned recognition mechanism, the *MakeCoffee* ADL can be first recognized. Suppose the inhabitant has a special way of making coffee that is modeled in the *ABox* as *myWayOfMakeCoffee*. Further *myWayOfMakeCoffee* is described by *hasContainer(cup)*, *hasAddings(milk)*, *hasHotDrinkType(coffee)*, *hasAddings(sugar)*, and *hasHotWater(hotWater)*. Through comparison, an assistive agent can infer that sugar and hot water are needed in order to complete the ADL. As the matching happens at the instance level, i.e., based on the way the user performs the activity and what actually happened, it can provide users with personalized assistance.

6 Experiment and Evaluation

6.1 Implementation and Deployment

The proposed approach has been implemented in a feature-rich context-aware assistive system as shown in Fig. 7 (the front-end interface [33]). The system was developed using

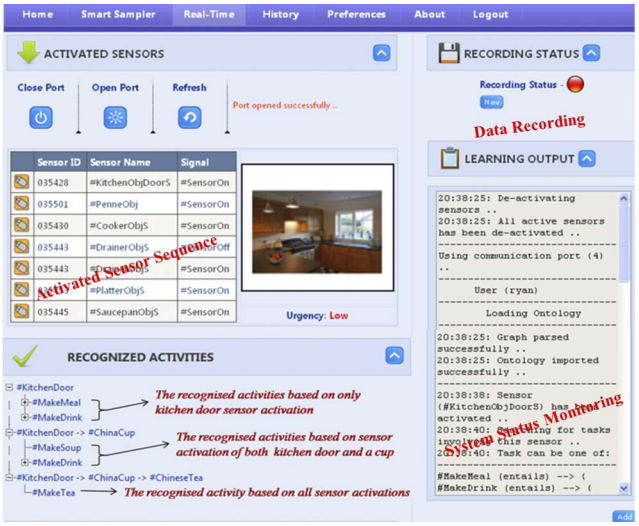

Fig. 7. The system interface in real-time mode.

C#, ASP.NET, Ajax, and Silverlight support for audio and graphical user experience. The creation, management, and query of the semantic data were handled using the SemWeb semantic technologies for C# [34] and SPARQL [35]. Semantic reasoning was implemented using the Euler inference engine [36].

To support the rich functionalities and features, the system provides a set of configuration tools, multiple graphical views, and quick-access function buttons (refer to Fig. 7). These tools are used to import activity and context ontologies, specify reasoning, and learning parameters, select the modality of audio reminder, configure hardware, e.g., communication ports, and define event priorities and user activity profiles. The views are used to show the deployed sensor network in the environment, display the sequence of activated objects, output a temporal trace of events during the system operation and present the recognized activity within a tree-like activity hierarchy. The function buttons allow a user to initiate operations, e.g., to record sensor activations in XML files or to put the system into the simulation mode using previously recorded sensor activations in a XML file.

The system is deployed in a computer in a smart home laboratory [37]. The spec of the computer used was as follows: Dell Optiplex755, Intel Duo CPU E6750 2.66 GHZ, 3.25 GB RAM, Windows XP SP3. The SH environment used for experimentation contains various objects for performing ADLs. A number of sensors, including contact sensors, motion sensors, tilt sensors, and pressure sensors, all from the Tynetec ZP0532 series, are available. A Tynetec receiver (www.tynetec.co.uk) is used to collect sensor activation signals through wireless communication which is subsequently processed by the computer system.

When the system is in operation, it obtains real-time sensor activations from a designated communication port that is connected to an external Tynetec receiver. Each time a sensor is activated, it will aggregate the information with previously collected activated sensors to generate an activity description. The description is then fed to the reasoning engine to infer the potential activity against activity models and profiles. As an actor interacts with the

TABLE 2
Two Example Activity Specifications

Activities		Activity Specification (sequence of user-object interactions identified by sensors)
makeTea	Type 1	GoToKitchen, GetCup, GetTea, PourWater, GetMilk, GetSugar
	Type 2	GoToKitchen, GetCup, **PourWater, GetMilk, GetTea**, GetSugar
	Type 3	GoToKitchen, **GetCup***, GetTea, PourWater, GetMilk, GetSugar
BrushTeeth	Type 1	GoToBathroom, RunSink, GetToothbrush, GetToothpaste, GetMouthwash
	Type 2	GoToBathroom, **GetToothbrush, GetToothpaste, RunSink**, GetMouthwash
	Type 3	GoToBathroom, RunSink, GetToothbrush, **getSoap****, GetToothpaste, GetMouthwash

∗ *faulty sensors that do not fire;* ∗∗ *false or extra sensor reading.*

objects in sequence in real time, sensor activations are continuously fed into the system. As such, recognition operations are repeatedly performed to realize continuous progressive activity recognition. As can be seen in Fig. 7, the system can dynamically display the activated sensor sequence, the incrementally recognized activities, and system status and data.

6.2 Experimental Setup

Eight typical ADLs (refer to Table 3) were selected for the purposes of experimentation. For each activity, the required objects for performing the activity were identified and to each of them an appropriate type of sensor was attached. For example, a tilt sensor was attached to a kettle for detecting the action of pouring water. For each activity, we specify how it is performed based on domain knowledge. It is specified as a sequence of user-object interactions. The interaction is detected by sensors when a user uses the objects to perform the activity. For example, the activity "making tea" is specified as the sequence "go to kitchen, take a cup, take a tea bag, add hot water, add milk, and add sugar." These activities and all the sensors are modeled in ontologies and represented in OWL, which are uploaded into the system during system startup.

In order to test not only functionalities but also scalability and robustness, each activity was designed to be performed in three different ways, leading to three types of activity specification. Table 2 shows two selected activities, each with three types of specification. The Type 1 activity specification can be viewed as the "standard" way of performing a specific activity. The experiment using the Type 1 activity specification is mainly aimed to test the functionalities and provide a baseline of activity recognition.

The Type 2 activity specification is obtained by deliberately changing the sequence of the objects being interacted with by a user (refer to Table 2). The main objective of this experiment is to test system scalability, i.e., if the system can still recognize the activity even if the way (the sequence of the objects used) of performing the activity has changed. The Type 3 activity specification is obtained by deliberately introducing sensor noise (uncertainty) to the Type 1 specification. We simulate a faulty (fails to activate) sensor by omitting a user-object interaction and a false sensor reading by adding an irrelevant object interaction. The purpose of this experiment is to test system robustness under instances of information uncertainty.

Approximately 40 sensors were deployed in the experiment, including two tilt sensors, two pressure sensors, and

TABLE 3
Recognition Results of the 144 Activities

Activities		Actor1		Actor2		Actor3		Sum Y/N
		Exp1	Exp2	Exp1	Exp2	Exp1	Exp2	
Make Tea	TP1	Y	Y	Y	Y	Y	Y	6/0
	TP2	Y	Y	Y	Y	**N**	Y	5/1
	TP3	Y	Y	Y	**N**	**N**	**N**	3/3
Brush Teeth	TP1	Y	Y	Y	Y	Y	Y	6/0
	TP2	Y	Y	Y	Y	Y	Y	6/0
	TP3	Y	Y	Y	Y	Y	Y	6/0
Make Coffee	TP1	Y	Y	Y	Y	Y	Y	6/0
	TP2	Y	Y	**N**	Y	**N**	Y	4/2
	TP3	Y	Y	Y	Y	Y	Y	6/0
Have Bath	TP1	Y	Y	Y	Y	Y	Y	6/0
	TP2	Y	Y	Y	Y	Y	Y	6/0
	TP3	Y	Y	Y	Y	Y	Y	6/0
Watch TV	TP1	Y	Y	Y	Y	Y	Y	6/0
	TP2	Y	Y	Y	Y	Y	Y	6/0
	TP3	Y	Y	Y	Y	Y	Y	6/0
MakeChocolate	TP1	Y	Y	Y	Y	Y	Y	6/0
	TP2	Y	Y	Y	Y	Y	Y	6/0
	TP3	Y	Y	Y	Y	Y	Y	6/0
Make Pasta	TP1	Y	Y	Y	Y	Y	Y	6/0
	TP2	Y	Y	Y	Y	**N**	Y	5/1
	TP3	Y	Y	**N**	Y	Y	Y	5/1
Wash Hands	TP1	Y	Y	Y	Y	Y	Y	6/0
	TP2	Y	Y	Y	Y	Y	Y	6/0
	TP3	Y	Y	Y	Y	Y	Y	6/0
Sum Y/N	All TPs	24/0	24/0	22/2	23/1	20/4	23/1	136/8

the remaining contact sensors. Each sensor was attached to one of the objects involved in these activities. On average, five objects (sensors) were used for each activity. The sliding time window was set to 5 minutes using the system configuration tool. This value is relatively long as in this experiment we do not set the duration properties for these selected activities. The recognition operation is set to be performed each time a sensor is activated, i.e., a user interacts with an object. This is deemed as necessary in order to provide just-in-time assistance in a real-assistive living scenario.

We selected three activities out of the eight activities, i.e., making tea, making coffee, and making pasta, to test and evaluate fine-grained activity recognition. For each activity, a special way of performing this specific activity, e.g., an activity profile was specified for each actor. For example, actor A had a profile "*ActorA_Preferred_MakeTea*" that specified the specific objects the actor uses for making tea.

Two types of experiment were designed and conducted. In the first experiment, the actor only used the objects specified by the preferred activity model. This experiment aims to test how accurate the system recognizes and further provides fine-grained assistance to users according to their preferences. It is assumed that when a user activity preference is available, the system should be able to remind the user, not just the type of action, e.g., adding milk, but also the specific type of object to be used, e.g., adding the semiskimmed milk. In the second experiment, the actor will perform the activity but replace one of the objects with the same type but a different object. This experiment aims to test how the system reacts to a noisy activity (information

uncertainty). In this case, the noise is of the same type but a different object. For example, a user prefers to use *Whole-Milk* for making tea, but the actor actually uses the semiskimmed milk.

6.3 Experimental Procedure

Three male actors with ages 25, 35, and 45 took part in the experiments. Each of the participants performed $3\ (\text{type}) \times 8\ (\text{activity}) = 24$ activities in terms of the activity specification for two rounds. This produced a total of $24\ (\text{activity scenarios}) \times 2\ (\text{rounds}) \times 3\ (\text{actors}) = 144$ activities being performed in the experiment.

For each activity, the experiment was carried out as follows: an actor was vocally prompted by a human evaluator to start performing an activity by following the object sequence of the specified activity scenario. The evaluator observed and recorded the actor's action and the recognition results of the system in an experiment data sheet. This provided the ground truth about individual user-object interactions and the activity for later evaluation. The interval between two consecutive actions was set to approximately 30 seconds. The exact timing of each user-object interaction and the recognition result was recorded by the system logger from which the runtime of each recognition operation can be extracted. In addition to an in-memory buffer for holding sensor activations within a time window, the system provides two automatic recording facilities to store sensor data permanently. The first was to record all sensor activations into a structured XML document representing all user-object interactions during the recording sessions, including sensor information and temporal data. The collected data were mainly used for advanced data processing, e.g., activity model learning, and user profile learning. The second method was similar to a server logger. It recorded everything from the start of the system, including sensor activations, the recognition results at each step, error messages, warning messages (e.g., a sensor battery level-low warnings). The recorded information was archived in a text file. The logging function provided a tool for system debugging and results checking.

For the fine-grained activity recognition experiment, two male actors each performed $2\ (\text{type}) \times 3\ (\text{activity}) = 6$ activities in terms of the activity specifications, producing a total of $6\ (\text{activity scenarios}) \times 2\ (\text{actors}) = 12$ activities. The procedure is the same as described above.

6.4 Results and Discussions

6.4.1 Activity Recognition Accuracy

Table 3 shows the recognition results of the 144 activities, where TP refers to the type of activity, Exp1 and Exp2 refer to the two rounds of experiments, respectively, Y and N denote the success and failure of activity recognition, and Sum in each row and column refers to the number of Y and N for a particular type of activity and a particular actor, respectively. In the analysis, activity recognition is deemed successful the first time an activity is correctly recognized. The rationale is that once an activity is recognized, there is no need for further recognition effort for the present activity until the next activity starts. The activity recognition system will pass the recognized activity to an assistive system, which will subsequently use the standard activity model to help users to complete the activity.

TABLE 4
Activity Recognition Accuracy (Percent)

Activity \ Actor	Actor 1	Actor 2	Actor3	Recognition Accuracy(Aggr.)
Type 1	100	100	100	100
Type 2	100	93.75	81.25	91.66
Type 3	100	87.5	87.5	91.66
Recognition Accuracy(Aggr.)	100	93.75	89.58	Overall: 94.44

The metric used for evaluation is recognition accuracy, defined as the percentage of the correctly recognized activities against the total activities of a specific experimental scenario. Table 4 shows recognition accuracies for a combination of scenarios. As can be seen from Table 4, recognition accuracies for type 1, 2, and 3 activity scenarios are 100, 91.66, and 91.66 percent, respectively, with the overall recognition accuracy of 94.44 percent. The accuracies for type 2 and 3 experiments are lower than the type 1 experiment. This was expected given that type 1 activities are performed according to the way they had been designed in the activity model. If we use type 1 experiments as a benchmark, then the results (91.66 percent) from type 2 experiments where activities are performed in an order different from that specified in the activity model, and type 3 (91.66 percent) experiment where various levels of sensor noise are introduced, proves that the approach is scalable, i.e., adaptive to different activity styles, and robust, i.e., resilient to noise (information uncertainty). Given that this is real-time continuous activity recognition with incomplete sensor data, we believe the results are quite impressive.

We also analyzed activity recognition accuracies for individual actors. The goal is to evaluate how much different actors affect the recognition performance of the system. In a real world environment, the system will be used by different users, the consistency of the system performance is therefore important. As can be seen from Table 4, the recognition accuracies for the three actors were 100, 93.75, and 89.58 percent, respectively, with small variations. This suggests that the system performance is consistent and stable in real-world contexts.

6.4.2 Real-Time System Performance

Real-time continuous activity recognition requires that the recognition operation occurs periodically during the performance of an activity. As such, the time taken for each recognition operation is critical for system responsiveness. We use the Time per Recognition Operation (TpRO) as the metric to evaluate the real-time system performance. The TpRO is defined as the interval in second(s) from the time a sensor is activated until the time the system produces a recognized activity. As can be seen from Table 5, the average TpROs for each activity ranges from 2.2 to 3.0 sec with the overall average TpRO roughly as 2.5 sec. Given the nature of ADLs, i.e., necessary object movement between locations, and the generic characteristics of users of assistive living, i.e., ageing people or cognitively impaired like dementia patients, the TpRO from the experiment proves

TABLE 5
The TpROs for All the Eight Activities Performed by Actor2 in Experiment2

Activity	Make Tea	Make Coffee	Make Chocolate	Make Pasta	Wash Hands	Brush Teeth	Have A Bath	Watch TV
Objects	5	6	6	7	4	6	5	4
TpRO	2.4	2.2	2.5	2.4	3	2.7	2.6	2.5

that the system can efficiently and effectively respond to real-world user-object interactions, thus offering real-time continuous recognition.

By analyzing TpROs for different activities, we also test the hypothesis that complex activities involving more user-object interactions will need more time for subsumption classification reasoning, thus higher TpRO to complete recognition operation. Nevertheless, the TpRO results in Table 5 disprove this hypothesis since the 7-object activity *Make Pasta* have a TpRO of 2.4 sec and the 3-objectt *Wash Hands* had a TpRO of 3.0 sec. The reason for this counter-intuitive finding may be relevant to other factors of affecting semantic reasoning, e.g., the location of an activity model in the hierarchical tree structure of activity ontologies, and also the number of similar activities, i.e., requiring more effort to distinguish from each other. While further systematic experimentation is required for an in-depth examination of this finding, the generic level and range of TpRO has proved that the system is applicable in real-world scenarios.

6.4.3 Fine-Grained Activity Recognition Accuracy

Table 6 shows the recognition results of the 12 fine-grained activities. Here, activity recognition is deemed as successful if it meets the following two criteria. The first is that the activity should be initially correctly recognized at some stage during the activity performance as a generic activity, i.e., coarse-grained recognition. The second is that once the activity is recognized, the system can discover the user's profile and remind users the specific objects they could use for performing the activity. Table 6 shows that the system achieves 100 percent recognition accuracy for the type 1 experiment. For type 2 experiment, the success of recognition needs to meet another criterion, i.e., when the same type but different object from the preferred object is used, the system should be able to recommend the preferred object. Again, the system achieves 100 percent recognition accuracy.

TABLE 6
Fine-Grained Activity Recognition Results

Activities		Actor1	Actor2
Make Tea	TP1	Y	Y
	TP2	Y	Y
Make Pasta	TP1	Y	Y
	TP2	Y	Y
Make Coffee	TP1	Y	Y
	TP2	Y	Y

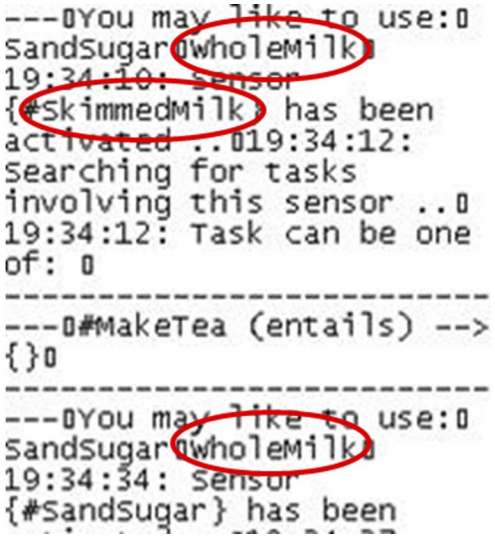

Fig. 8. A log trace fragment.

Fig. 8 shows a fragment of system logs that contain sensor activation and activity recognition traces. The highlighted trace text (by the three circles) indicates that the system recommended using *WholeMilk* for making tea. While the *SkimmedMilk* was actually used, the system still recommended the use of the preferred *WholeMilk*. In a real use case, the system can issue a warning to stop the use of *SkimmedMilk*. This is the feature of assistance provisioning that is not the focus of this paper.

6.4.4 User-Object Interaction Recognition

Throughout the experiment of 144 activities, a total of 804 user-object interactions were performed. We used the User-object Interaction Recognition Accuracy (UoIR), defined as the system's correctly captured interactions against the total occurred interactions, as the metric to evaluate the reliability and performance of the activity monitoring mechanism. This metric takes into account not only unfired or misread interactions caused by faulty sensors but also those circumstances caused by wireless receivers, communication, and system sampling, and conversion mechanisms. As such, it is more accurate to reflect the system monitoring performance. Table 7 shows the UoIR for different types of sensors with an overall average UoIR of 96.89 percent. This proves the monitoring and acquisition mechanism of the system as being very reliable.

TABLE 7
Interaction Recognition Rate

Sensor Types	Total Interaction	Captured Interaction	Accuracy (%)
Contact	624	611	97.92
Tilt	126	119	94.44
Pressure	36	32	88.89
Sound	18	17	94.44

A detailed examination of the UoIR of individual sensors indicated that the contact sensors had the highest UoIR (97.92 percent) while the pressure sensor had the lowest UoIR (88.89 percent). This is consistent with our observation that pressure sensors tend to be very sensitive to the weight variation and the change of sitting posture or other movements can easily generate noise activations.

7 Conclusions

In this paper, we proposed a knowledge-driven approach to activity recognition based on ontological modeling and semantic reasoning. We have analyzed the nature and characteristics of ADLs upon which we argue the necessity of domain knowledge for pattern recognition using multisource sensor data. Following knowledge engineering practices, we have developed context and ADL ontologies for the SH. We have conceived and designed an agent-based integrated system architecture to illustrate the realization of the proposed approach. The compelling feature of the system is the unified ontological modeling and representation for both sensor data and activities, which not only facilitates domain knowledge reuse but also allows the exploitation of semantic reasoning for activity recognition. In particular, we developed a novel activity recognition algorithm that allows continuous activity recognition at multiple levels of abstraction, thus enabling course-grained and fine-grained activity assistance. The mechanism for evolving initial ontological activity models through learning provides a way of marrying the strengths of traditional data-driven approaches with knowledge-driven practices, making our approach flexible, applicable, and scalable in terms of rapid system development and deployment.

We have implemented our approach in a feature-rich recognition and assistance system. The system has been tested and validated in both real-world and simulated activity scenarios. While the full evaluation of this approach awaits further investigation and user feedback, our initial results have demonstrated the approach is viable and robust in real-world use cases. The approach and the example application scenario have both been developed in a specific application context, namely that of SH-based assistive living. We have not seen any reasons to prevent this approach from being applied to other types of sensorized applications.

The importance of domain knowledge is nowhere more apparent than in the field of assistive living—every individual has their own way of performing activities. We have demonstrated this with respect to one aspect of expertise, namely domain knowledge-driven activity recognition from multisource sensor data. Our future work will focus on large-scale experimentation of a diversity of activity scenarios, multiuser concurrent activity recognition and handling of temporal information such as sequence and duration.

Acknowledgments

The authors gratefully acknowledge the contributions from Ahmad Al-Bashrawi, Ryan Goldblatt, and Mark Bettie for the implementation, deployment, and testing of the system.

References

[1] M. Chan, D. Estève, and C. Escriba, "A Review of Smart Homes—Present State and Future Challenges," *Computer Methods and Programs in Biomedicine,* vol. 91, no. 1, pp. 55-81, 2008.

[2] D. Cook, H. Hagras, V. Callaghan, and A. Helal, "Making Our Environments Intelligent," *J. Pervasive and Mobile Computing,* vol. 5, pp. 556-557, 2009.

[3] S. Helal, W. Mann, H. El-Zabadani, J. King, Y. Kaddoura, and E. Jansen, "The Gator Tech Smart House: A Programmable Pervasive Space," *Computer,* vol. 38, no. 3, pp. 50-60, Mar. 2005.

[4] D.J. Cook and P. Rashidi, "Keeping the Resident in the Loop: Adapting the Smart Home to the User," *IEEE Trans. Systems, Man, and Cybernetics J., Part A,* vol. 39, no. 5, pp. 949-959, Sept. 2009.

[5] The Netwell Centre, Technologies for Ageing in Place, http://netwellcentre.ie, 2012.

[6] D. Cook and S.K. Das, "How Smart Are Our Environments? An Updated Look at the State of the Art," *J. Pervasive and Mobile Computing,* vol. 3, no. 2, pp. 53-73, 2007.

[7] L. Chen, C. Nugent, M. Mulvenna, D. Finlay, and X. Hong, "A Logical Framework for Behaviour Reasoning and Assistance in a Smart Home," *Int'l J. Assistive Robotics and Mechatronics,* vol. 9, no. 4, pp. 20-34, 2008.

[8] M. Philipose, K.P. Fishkin, M. Perkowitz, D.J. Patterson, D. Fox, H. Kautz, and D. Hahnel, "Inferring Activities from Interactions with Objects," *IEEE Pervasive Computing,* vol. 3, no. 4, pp. 50-57, Oct.-Dec. 2004.

[9] W. Geyer, H. Richter, and G.D. Abowd, "Towards a Smarter Meeting Record—Capture and Access of Meetings Revisited," *Multimedia Tools and Applications,* vol. 27, no. 3, pp. 393-410, 2005.

[10] Z. Yu and Y. Nakamura, "Smart Meeting Systems: A Survey of State-of-the-Art and Open Issues," *ACM Computing Surveys,* vol. 42, no. 2, Article 8, 2010.

[11] J. Hoey and J. James, "Little Value-Directed Human Behavior Analysis from Video Using Partially Observable Markov Decision Processes," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 29, no. 7, pp. 1118-1132, July 2007.

[12] T.B. Moeslund, A. Hilton, and V. Krüger, "A Survey of Advances in Vision-Based Human Motion Capture and Analysis," *Computer Vision Image Understanding,* vol. 104, no. 2, pp. 90-126, 2006.

[13] L. Fiore, D. Fehr, R. Bodor, A. Drenner, G. Somasundaram, and N. Papanikolopoulos, "Multi-Camera Human Activity Monitoring," *J. Intelligent and Robotic Systems,* vol. 52, no. 1, pp. 5-43, 2008.

[14] P. Turaga, R. Chellappa, V.S. Subrahmanian, and O. Udrea, "Machine Recognition of Human Activities: A Survey," *IEEE Trans. Circuits and Systems for Video Technology,* vol. 18, no. 11, pp. 1473-1488, Nov. 2008.

[15] S.W. Lee and K. Mase, "Activity and Location Recognition Using Wearable Sensors," *IEEE Pervasive Computing,* vol. 1, no. 3, pp. 24-32, 2002.

[16] J. Parkka, M. Ermes, P. Korpipaa, J. Mantyjarvi, J. Peltola, and I. Korhonen, "Activity Classification Using Realistic Data from Wearable Sensors," *IEEE Trans. Information Technology in Biomedicine,* vol. 10, no. 1, pp. 119-128, Jan. 2006.

[17] D. Sanchez and M. Tentori, "Activity Recognition for the Smart Hospital," *IEEE Intelligent Systems,* vol. 23, no. 2, pp. 50-57, Mar./Apr. 2008.

[18] L. Bao and S. Intille, "Activity Recognition from Userannotated Acceleration Data," *Proc. Int'l Conf. Pervasive Computing,* pp. 1-17, 2004.

[19] O. Brdiczka, J.L. Crowley, and P. Reignier, "Learning Situation Models in a Smart Home," *IEEE Trans. Systems, Man and Cybernetics—Part B: Cybernetics,* vol. 39, no. 1, pp. 56-63, Feb. 2009.

[20] J. Hoey and P. Poupart, "Solving POMDPs with Continuous or Large Discrete Observation Spaces," *Proc. Int'l Joint Conf. Artificial Intelligence,* pp. 1332-1338, 2005.

[21] B. Bouchard and S. Giroux, "A Smart Home Agent for Plan Recognition of Cognitively-Impaired Patients," *J. Computers,* vol. 1, no. 5, pp. 53-62, 2006.

[22] D. Preuveneers, J. Van den Bergh, and K. De Bosschere, "Towards an Extensible Context Ontology for Ambient Intelligence," *Proc. Second European Symp. Ambient Intelligence (EUSAI '04),* pp. 148-159, 2004.

[23] A.B. James, "Activities of Daily Living and Instrumental Activities of Daily Living," *Willard and Spackman's Occupational Therapy,* E.B. Crepeau, E.S. Cohn, B.B. Schell, eds., pp. 538-578, Lippincott, Williams and Wilkins, 2008.

[24] V. Wadley, O. Okonkwo, M. Crowe, and L.A. Ross-Meadows, "Mild Cognitive Impairment and Everyday Function: Evidence of Reduced Speed in Performing Instrumental Activities of Daily Living," *Am. J. Geriatr Psychiatry*, vol. 16, no. 5, pp. 416-424, 2007.
[25] U. Akdemir and P. Turaga, "An Ontology Based Approach for Activity Recognition from Video," *Proc. 16th ACM Int'l Conf. Multimedia*, pp. 709-712, 2008.
[26] E.M. Tapia and S. Intille, "Real-Time Recognition of Physical Activities and Their Intensities Using Wireless Accelerometers and a Heart Rate Monitor," *Proc. Int'l Symp. Wearable Computers (ISWC)*, pp. 1-4, 2007.
[27] N. Yamada, K. Sakamoto, G. Kunito, Y. Isoda, K. Yamazaki, and S. Tanaka, "Applying Ontology and Probabilistic Model to Human Activity Recognition from Surrounding Things," *IPSJ Digital Courier*, vol. 3, pp. 506-517, 2007.
[28] G. Okeyo, L. Chen, H. Wang, and R. Sterritt, "Ontology-Enabled Activity Learning and Model Evolution in Smart Homes," *Proc. Seventh Int'l Conf. Ubiquitous Intelligence and Computing*, 2010.
[29] F. Baader, I. Horrocks, and U. Sattler, "Description Logics," *Handbook of Knowledge Representation*, F. van Harmelen, V. Lifschitz, and B. Porter, eds., Elsevier, 2007.
[30] D. Tsarkov and I. Horrocks, "FaCT++ Description Logic Reasoner: System Description," *Proc. Int'l Joint Conf. Automated Reasoning (IJCAR '06)*, pp. 292-297, 2006.
[31] I. Horrocks and U. Sattler, "A Tableau Decision Procedure for SHOIQ," *J. Automated Reasoning*, vol. 39, no. 3, pp. 249-276, 2007.
[32] B. Motik, R. Shearer, and I. Horrocks, "Hypertableau Reasoning for Description Logics," *J. Artificial Intelligence Research*, vol. 36, pp. 165-228, 2009.
[33] L. Chen and C.D. Nugent, "Ontology-Based Activity Recognition in Intelligent Pervasive Environments," *Int'l J. Web Information Systems*, vol. 5, no. 4, pp. 410-430, 2009.
[34] Semantic Web/RDF Liberary for C#.NET, http://razor.occams.info/code/semweb/, 2012.
[35] SPARQL Query Language for RDF, http://www.w3.org/TR/rdf-sparql-query/, 2012.
[36] Euler Proof Mechanism, http://www.agfa.com/w3c/euler/, 2012.
[37] C.D. Nugent, M. Mulvenna, X. Hong, and S. Devlin, "Experiences in the Development of a Smart Lab," *The Int'l J. Biomedical Eng. and Technology*, vol. 2, no. 4, pp. 319-331, 2009.
[38] The Protégé Framework, http://protege.stanford.edu/, 2012.

Liming Chen received the BSc and MSc degrees in computing engineering from Beijing Institute of Technology, China, and DPhil degree in artificial intelligence from De Montfort University, United Kingdom. He is a lecturer at the School of Computing and Mathematics, University of Ulster, United Kingdom. His current research interests include the semantic technologies, ontology-enabled knowledge management, intelligent agents, information/knowledge fusion and reasoning, semantic sensor networking, assistive technologies and their applications in smart homes and intelligent environments.

Chris D. Nugent received the Bachelor of Engineering degree in electronic systems and DPhil degree in biomedical engineering both from the University of Ulster. He is a professor at the School of Computing and Mathematics, University of Ulster, United Kingdom. He currently holds the position of professor of biomedical engineering within the School of Computing and Mathematics at the University of Ulster. His research addresses themes of technologies to support independent living, medical decision support systems, and the development of Internet-based healthcare models.

Hui Wang received the BSc degree in computer science and MSc degree in artificial intelligence both from Jilin University, China, and the PhD degree in artificial intelligence from the University of Ulster. He is a reader at the School of Computing and Mathematics, University of Ulster, United Kingdom. His research interests include pattern recognition, machine learning, data/text mining, financial data mining, uncertainty reasoning, spatial reasoning, and information retrieval.